第六章习题

（1）在一个图中，所有顶点的度数之和等于图的边数的（ ）倍。

A．1/2 B．1 C．2 D．4

答案：C

（2）在一个有向图中，所有顶点的入度之和等于所有顶点的出度之和的（ ）倍。

A．1/2 B．1 C．2 D．4

答案：B

解释：有向图所有顶点入度之和等于所有顶点出度之和。

（3）具有n个顶点的有向图最多有（ ）条边。

A．n B．n(n-1) C．n(n+1) D．n2

答案：B

解释：有向图的边有方向之分，即为从n个顶点中重复选取2个顶点有序排列，结果为n(n-1)。

（4）n个顶点的连通图用邻接距阵表示时，该距阵至少有（ ）个非零元素。

A．n B．2(n-1) C．n/2 D．n2

答案：B

所谓连通图一定是无向图，有向图叫强连通图

连通n个顶点,至少只需要n-1条边就可以了，由于无向图的每条边同时关联两个顶点,因此邻接矩阵中每条边被存储了两次（也就是说是对称矩阵）,因此至少有2(n-1)个非零元素

（5）G是一个非连通无向图，共有28条边，则该图至少有（ ）个顶点。

A．7 B．8 C．9 D．10

答案：C

解释：8个顶点的无向图最多有8\*7/2=28条边，再添加一个点即构成非连通无向图，故至少有9个顶点。

（6）若从无向图的任意一个顶点出发进行一次深度优先搜索可以访问图中所有的顶点，则该图一定是（ ）图。

A．非连通 B．连通 C．强连通 D．有向

答案：B

解释：即从该无向图任意一个顶点出发有到各个顶点的路径，所以该无向图是连通图。

（7）下面（　）算法适合构造一个稠密图G的最小生成树。

A． Prim算法 B．Kruskal算法 C．Floyd算法 D．Dijkstra算法

答案：A

解释：Prim算法适合构造一个稠密图G的最小生成树，Kruskal算法适合构造一个稀疏图G的最小生成树。

（8）用邻接表表示图进行广度优先遍历时，通常借助（ ）来实现算法。

A．栈 B. 队列 C. 树 D．图

答案：B

解释：广度优先遍历通常借助队列来实现算法，深度优先遍历通常借助栈来实现算法。

（9）用邻接表表示图进行深度优先遍历时，通常借助（ ）来实现算法。

A．栈 B. 队列 C. 树 D．图

答案：A

解释：广度优先遍历通常借助队列来实现算法，深度优先遍历通常借助栈来实现算法。

（10）深度优先遍历类似于二叉树的（ ）。

A．先序遍历 B．中序遍历 C．后序遍历 D．层次遍历

答案：A

（11）广度优先遍历类似于二叉树的（ ）。

A．先序遍历 B．中序遍历 C．后序遍历 D．层次遍历

答案：D

（12）图的BFS生成树的树高比DFS生成树的树高（ ）。

A．小 B．相等 C．小或相等 D．大或相等

答案：C

解释：对于一些特殊的图，比如只有一个顶点的图，其BFS生成树的树高和DFS生成树的树高相等。一般的图，根据图的BFS生成树和DFS树的算法思想，BFS生成树的树高比DFS生成树的树高小。

（13）已知图的邻接矩阵如图6.30所示，则从顶点v0出发按深度优先遍历的结果是（ ）。



​ 图6.30 邻接矩阵

答案：C

因为是深度优先,找到与顶点0直接相连的结点,由邻接矩阵知道是顶点1（多个相邻节点取第一个找到的未遍历到的结点）,然后再在邻接矩阵中找与顶点1直接相连的结点,得到顶点3.相同方法找到后续结点为：顶点4,顶点2.因为顶点2的相连结点都已被遍历,所以退回到顶点4继续遍历,遍历到顶点5,然后是顶点6

（14）已知图的邻接表如图6.31所示，则从顶点v0出发按广度优先遍历的结果是（ ），按深度优先遍历的结果是（ ）。

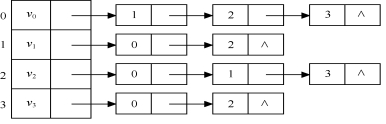


图6.31 邻接表

A．0 1 3 2 B．0 2 3 1 C．0 3 2 1 D．0 1 2 3

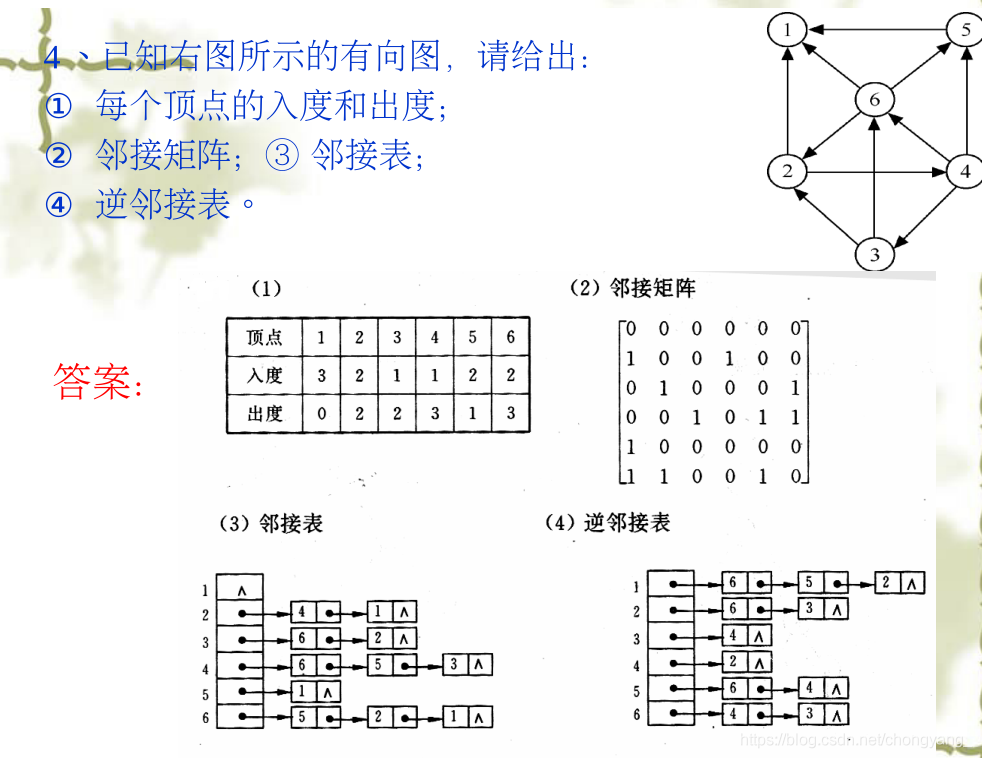
答案：D、D

广度：访问V0，依次访问其未访问的邻接顶点（顺着链表）

深度：首先邻的结点 1，然后找到 2，然后找到 3

（15）下面（ ）方法可以判断出一个有向图是否有环

A．深度优先遍历 B．拓扑排序 C．求最短路径 D．求关键路径



答案：B

————————————————

版权声明：本文为CSDN博主「念远\_」的原创文章，遵循CC 4.0 BY-SA版权协议，转载请附上原文出处链接及本声明。

原文链接：https://blog.csdn.net/chongyang\_/article/details/109609180